Looking for the event list

Eduardo Miranda, Buenos Aires, Argentina

1. Introduction

In modern structured analysis [1,2] a list of all the external and temporal stimuli to which the system must respond to fulfil its purpose is constructed. This list, called the event list, is used to establish system boundaries, as partition criteria and as a starting point for requirement analysis.

How to build such list is a key issue and is actually a question only partially answered. Ward [2,3] suggests a brainstorming approach to its construction. He advices work to be done by a group and not to be considered complete until every conceivable event to which the system might respond, no matter how far-fetched, has been examined and accepted or rejected. To discover new events, system developers can apply the following questions to an event list already founded:

- Are there any variations of these events that are significant ?
- Are the opposite, or negative, of these events of interest to the system ?
- Are there any events that must precede/follow these events ?
- What if the events fail to occur ?

This paper presents a complementary strategy based on the knowledge of the state of the objects with which the system has to deal.

2. The state transition approach to event finding

At the beginning of requirement analysis, and knowing the system purpose, it will be possible to identify objects that will be part of the system essential memory. These objects usually evolve through a series of states which the user is aware of. The question is in response to what, does a state change take place ?. The answer is simple: changes are due to events for which the system must provide a preplanned response.

As states are in the application domain, the user knows them, the originating events should be there too. Experience shows that the state concept is more familiar to him/her than the event concept. Thus, studying the states to find out the events facilitates the user analyst-dialogue.

As an example of how the method works, a simplified library system is presented. Figure 1 depicts a state transition diagram* of the five possible states a book can be in, from a librarian point of view.

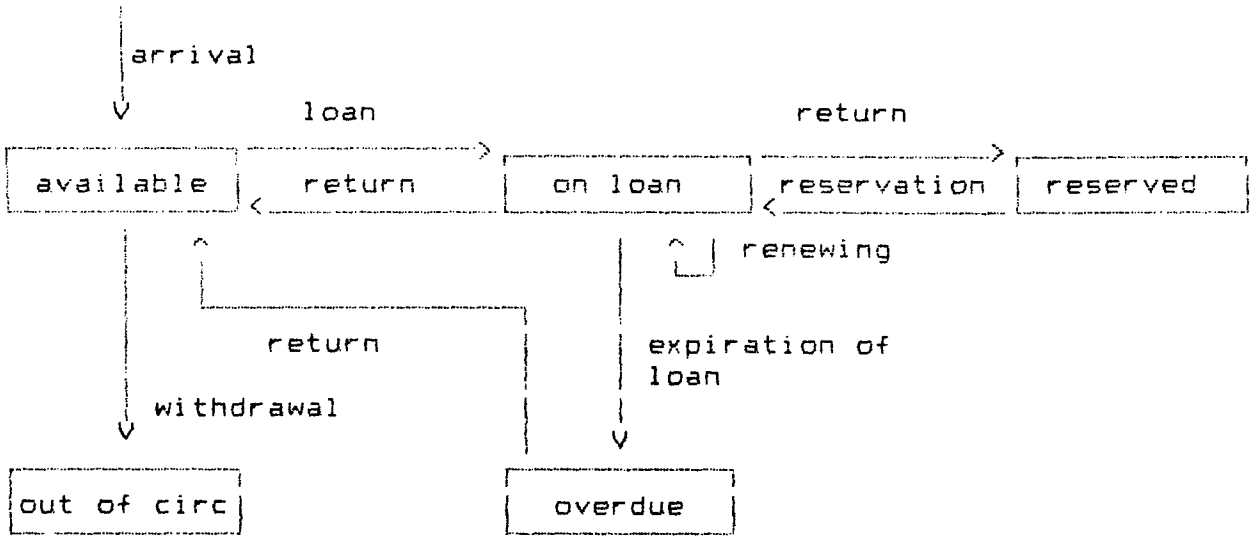

fig. 1 State transition diagram for a book in the library system

The event list derived from this state transition diagram is shown in Table 1. This list can be augmented applying the procedure to other objects in the system, looking for what causes relations among those objects to be established and using the brainstorming approach outlined at the beginning of this work.

3. Summary

The state transition approach complements the pure brainstorming strategy as a starting point in finding the event list. Its graphic representation by means of states transition diagram and its intuitive nature facilitates the user-analyst communication.

This technique serve both as an explanation and as a checking tool. Attaching the corresponding process name to each transition helps to guide a walkthrough of the system behavioral model.

* Note: Although the state transition diagram is not an essential part of the method, it integrates nicely with the other tools of structured analysis.

Event		Response
A book is acquired	E	updated catalog
A library member borrows a book	E	updated book status updated member's on-loan list
A library member returns a book	E	updated book status updated member's on-loan list
A library member renews a loan	E	updated book return date
A library member asks for a book already borrowed	E	updated book status updated waiting list
A loan period expires	T	overdue notice updated member status
A book is taken out of circulation	E	updated book status

E - External, T - Temporal

Table 1 - Event list derived from the state transition diagram

4. Acknowledgments

I wish to thank Raul Martinez, Pilar Barrio and to my coleagues at VCC for their helpful comments on an earlier version of this article.

5. References

1. Essential System Analysis, McMenamin and Palmer, Prentice-Hall.

2. System Development Without Pain, Ward, Prentice-Hall.

3. Structure Development for Real-Time Systems, Ward and Mellor, Prentice-Hall.